日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日
Date of Application: ２００３年１１月１７日

出 願 番 号
Application Number: 特願２００３－３８６０９１

[ST. 10/C]: [ＪＰ２００３－３８６０９１]

出 願 人
Applicant(s): 株式会社日立製作所

U.S. Appln. Filed 1-23-04
Inventor: S. Oshima et al
Mattingly Stanger & Malur
Docket ASA-1161

２００４年 １月 ８日

特許庁長官
Commissioner,
Japan Patent Office 今井康夫

出証番号 出証特２００３－３１０９６８６

【書類名】　特許願
【整理番号】　K03007621A
【あて先】　特許庁長官殿
【国際特許分類】　G06F　9/00
【発明者】
　【住所又は居所】　神奈川県川崎市麻生区王禅寺１０９９番地　株式会社日立製作所システム開発研究所内
　【氏名】　大島　訓
【発明者】
　【住所又は居所】　愛知県尾張旭市晴丘町池上１番地　株式会社日立製作所情報機器事業部内
　【氏名】　冨田　理
【発明者】
　【住所又は居所】　神奈川県川崎市麻生区王禅寺１０９９番地　株式会社日立製作所システム開発研究所内
　【氏名】　木村　信二
【特許出願人】
　【識別番号】　000005108
　【氏名又は名称】　株式会社　日立製作所
【代理人】
　【識別番号】　100075096
　【弁理士】
　【氏名又は名称】　作田　康夫
【選任した代理人】
　【識別番号】　100100310
　【弁理士】
　【氏名又は名称】　井上　学
【手数料の表示】
　【予納台帳番号】　013088
　【納付金額】　21,000円
【提出物件の目録】
　【物件名】　特許請求の範囲　1
　【物件名】　明細書　1
　【物件名】　図面　1
　【物件名】　要約書　1

【書類名】特許請求の範囲
【請求項１】
アプリケーションプログラムと共有ライブラリとを同一の仮想空間に配置し、かつ、アプリケーション毎に固有の仮想空間を有するようメモリ管理を行うオペレーティングシステムを実行する計算機において、
前記アプリケーションによる前記共有ライブラリへの関数呼出を注入共有ライブラリによってフックし、
前記注入共有ライブラリが前記メモリのカーネル領域に格納されたモジュールを呼び出し、
前記モジュールに設定された情報に基づいて該モジュールが処理を行い、その後、前記モジュールが前記共有ライブラリの関数を呼び出すことを特徴とする共有ライブラリ呼び出し方法。
【請求項２】
前記モジュールが、
前記フックされた前記共有ライブラリへの呼出の内容を前記カーネル領域内に記録することを特徴とする請求項１記載の共有ライブラリ呼び出し方法。
【請求項３】
前記モジュールが、
前記呼出の内容をあらかじめ定められたポリシに照らして監査し、
前記共有ライブラリの呼出の可否を決定することを特徴とする請求項２記載の共有ライブラリ呼び出し方法。
【請求項４】
前記呼び出しの可否の結果を前記カーネル領域内に記録することを特徴とする請求項３記載の共有ライブラリ呼び出し方法。
【請求項５】
前記呼出内容を前記カーネル領域内に記録する領域をプロセッサ毎に設けることを特徴とする請求項４記載の共有ライブラリ呼び出し方法。
【請求項６】
前記カーネル領域内に記録された呼出内容を取得する出力アプリケーションを有することを特徴とする請求項５記載の共有ライブラリ呼び出し方法。
【請求項７】
前記出力アプリケーションが前記モジュールの設定を変更するコマンドを発行し、
前記コマンドを、前記カーネル領域の所定の領域に格納し、
前記モジュールが前記所定の領域に格納されたコマンドに基づいて設定を変更することを特徴とする請求項６記載の共有ライブラリ呼び出し方法。
【請求項８】
前記出力アプリケーションが前記監査のポリシを設定するコマンドを発行し、
前記コマンドを、前記カーネル領域の所定の領域に格納し、
前記モジュールが前記所定の領域に格納されたコマンドに基づいて前記監査のポリシを設定することを特徴とする請求項７記載の共有ライブラリ呼び出し方法。

【書類名】明細書
【発明の名称】共有ライブラリに格納された外部公開関数の呼び出し方法
【技術分野】
【０００１】
本発明は、計算機で実行されるアプリケーションによる共有ライブラリに格納された外部公開関数の呼び出し方法及びその方法を実行する計算機に関する。
【背景技術】
【０００２】
一般の計算機は、ＣＰＵ、メモリ、ストレージ、通信装置、キーボード、マウス又はカード読取装置といった入力装置、ディスプレイ又はプリンタといった出力装置で構成される。
【０００３】
計算機上で動作するソフトウェア、具体的にはオペレーティングシステム（以下「ＯＳ」）、アプリケーション、共有ライブラリ等は、ストレージに格納されている。
【０００４】
尚、共有ライブラリとは、ストレージ１０３の記憶容量又はメモリ１０５の使用量を削減するため、ソフトウェア世代管理手続きを削減するため、又はソフトウェア開発コストを削減するために、種々のプログラムで実行される定型的な手続き（関数等）を集めたファイルあるいはモジュールである。この共有ライブラリは、複数のプログラムで共有されるため、アプリケーション等のプログラム本体とは別個に作成、管理、保存される。
【０００５】
共有ライブラリには、プログラムコンパイル時にプログラム本体と結合される静的結合ライブラリと、実行時に結合される動的結合ライブラリとがある。
【０００６】
計算機では、図４に示すように、アプリケーションが使用する記憶空間を、アプリケーション領域１（４０１）、アプリケーション領域２（４０２）、アプリケーション領域３（４０４）のようにアプリケーション固有に持たせる場合が多い。ただし、全ての処理に共通して使用されるＯＳについては、アプリケーションとは別の記憶領域が割り当てられる。この領域をカーネル領域４０７という。通常、このカーネル領域４０７は、特別な権限が無いアプリケーションからは自由にアクセスされない。
【０００７】
これによって、各アプリケーションは他のアプリケーションが含む問題によって干渉されたり、破壊されたりすることを防ぐことができる。このとき、動的結合ライブラリは、ＯＳの処理によって、実行コード領域とプログラム実行中に変更されない定数領域が、共有ライブラリ領域４０３、４０５、４０６のように、複数のプロセス間で共有される、即ち、一つの共有ライブラリが複数のメモリ領域にマッピングされるようになっている場合が多く、この場合、メモリ使用量削減やプログラム起動時間の削減効果が期待できる。
【０００８】
また、アプリケーションからカーネル領域４０７へのアクセスを制限しているＯＳの場合、一般的に、アプリケーションはカーネル領域４０７へのアクセス要求（以下「システムコール」）を発行する際、カーネル領域４０７で実行されているＯＳ２０１等のプログラムに対して、ソフトウェア割り込みを発行する。
【０００９】
ところが、ＯＳによってはシステムコールをカーネル領域４０７へのソフトウェア割り込みとしてではなく、動的結合ライブラリとして実装している場合がある。具体的には、アプリケーションがカーネル領域４０７で実行されているＯＳを呼び出す場合、一旦共有ライブラリを呼び出し、その共有ライブラリ内でソフトウェア割り込みを発生させることで、カーネル領域４０７へアクセスするのである。
【００１０】
非特許文献１には、こうしたアプリケーションからの共有ライブラリの呼び出しを、割

り込みに関わるＯＳやアプリケーションと異なるプログラム（以下「外部プログラム」）によって捕捉（以下「フック」）し、その呼出内容を記録し、又はフィルタリングする方法として動的結合ライブラリへの注入という技術が開示されている。

【００１１】

注入は、図５に示すように、注入対象、すなわち呼び出しの対象となる共有ライブラリ（以下「注入対象共有ライブラリ」）５０３と同じ外部公開関数を有する注入共有ライブラリ５０２を作成し、注入対象共有ライブラリの代わりにアプリケーション５０１と結合（アプリケーションの記憶空間にマッピング）させる。さらに注入共有ライブラリは、アプリケーションの共有ライブラリ呼出に対する、記録、監査、拒否といった処理を行う。その処理が終了した後、注入共有ライブラリは、注入対象である共有ライブラリを呼び出すよう設計されるので、アプリケーションからはシステムが注入共有ライブラリなしで運用されている場合と同じように動作しているように見える。

【００１２】

【非特許文献１】Jeffrey Richter著 長尾高弘訳 Advanced Windows 改訂第３版 アスキー出版局。

【発明の開示】

【発明が解決しようとする課題】

【００１３】

従来技術では、共有ライブラリに対応する記憶領域４０３等が各アプリケーションの記憶領域４０１等の中に配置されるため、共有ライブラリから呼び出された関数等の変数が格納される記憶領域はアプリケーション毎の個別の記憶領域になるという問題がある。例えば、注入共有ライブラリによって共有ライブラリ呼び出しの記録をする場合、記録すべき変数等の情報が各アプリケーションに割り当てられた記憶空間に配置されていると、その情報を収集するために、計算機ではプロセス間通信や記憶空間切り替えを行って、目的の情報を有する記憶空間に仮想記憶空間を切り替える必要が発生し性能の低下を招く。またプログラムが複雑になることによって信頼性の低下を招く。

【００１４】

上記の不都合を解消するために、アプリケーションが配置されている記憶空間に各アプリケーションが共通に使用できる領域を設け、共有ライブラリで使用される変数等の情報を記録することが考えられる。しかし、時間の経過によって実行されるアプリケーションが切り替えられる可能性がある。この場合、一貫性をもった記録を作成するには、アプリケーション間でのロック機構を設ける必要がある。これは記録を頻繁に残さなければならない場合に性能低下につながり、また複数のロックを取得しなければならない場合、デッドロックに陥ってシステムが停止してしまう危険が発生するという問題がある。

【００１５】

さらに、共有ライブラリの動作を制御（変数を指定）することを考えた場合、共有ライブラリの変数領域はアプリケーション毎に個別に用意されるため、ファイル等共有領域を設けて、そこに動作制御用のコマンドを書き込み、共有ライブラリが定期的に共有領域を参照することによって、動作制御用のコマンドを受信するという構成にする必要がある。しかし、ファイル等の共有領域の読み込みは、システムコールに比べて非常に時間がかかるので、性能上問題があるうえ、システムの状態によってはファイルアクセスができない場合があるという問題もある。

【課題を解決するための手段】

【００１６】

本発明では、上記課題を解決するため、アプリケーションから共有ライブラリに格納された外部公開関数の呼び出しを注入共有ライブラリでフックする際に使用する注入共有ライブラリの共有領域や共有ライブラリへのコマンド伝達領域をカーネル領域に設ける。

【００１７】

又、注入共有ライブラリで外部公開関数の呼び出しを監査する際に使用される基準についての情報を格納する領域もカーネル領域に設ける。

【００１８】
更に、各アプリケーションの共有ライブラリからの外部公開関数の呼び出し履歴（以下「ログ」）を格納する領域も、カーネル領域に設ける。
【００１９】
本発明の他の特徴については、本明細書及び添付図面の記載により明らかにする。
【発明の効果】
【００２０】
共有ライブラリの外部公開関数へのアクセスを共有ライブラリの書換え無しにフックしてフィルタリングする際、一貫性のあるログを取得することが可能となる。
【発明を実施するための最良の形態】
【００２１】
以下、実施形態の概要について説明する。
本実施形態においては、計算機１０１は、ソフトウエアプログラムとして、共有ライブラリ、共有ライブラリと関連付けられ、アプリケーションに結合される注入共有ライブラリ、カーネル領域に配置され、注入共有ライブラリがログ記録、管理者からのコマンド受信、共有ライブラリ外部公開関数呼び出し監査などの処理を行うフィルタモジュール及び注入共有ライブラリがカーネル領域に設けられた共通領域（以下「カーネルモード共有領域」）へ出力した結果を収集し、ファイルへの出力、管理者への報告などを行う出力アプリケーションを備える。
【００２２】
注入共有ライブラリは、注入対象となる共有ライブラリが有する外部公開関数へのアクセスの一部または全部をフックする。
【００２３】
カーネルモード共有領域は、カーネル領域の拡張であっても、カーネルへの動的追加モジュールであっても、カーネルモードで動作するデバイスドライバで実現されても良い。また、カーネルモード共有領域内には共有ライブラリ呼び出しの監査に使用する監査ポリシが配置される場合もある。
【００２４】
出力アプリケーションは、定期的またはカーネルモード共有領域からの呼び出しに基づいて、カーネルモード共有領域内に記録された情報を特定のファイルに出力する、管理者に報告を行うといった処理を行う。また、システム管理者からの指示といった注入共有ライブラリへのコマンドは、出力アプリケーションがカーネルモード共有領域内のコマンド受信領域に対し、書き込みを行うことによって発信することとできる。
【００２５】
アプリケーションから共有ライブラリへの関数の呼び出しが発生すると、呼び出しは注入共有ライブラリにフックされ、注入共有ライブラリへと制御が移る。注入共有ライブラリでは、呼び出し時刻、呼び出しを行ったユーザ、呼び出しを行ったアプリケーション、呼び出し時に渡された引数といった情報を取得することが可能である。注入共有ライブラリでは、これらの情報のカーネルモード共有領域への記録を、カーネル領域で実行されているフィルタモジュールにシステムコールの形で依頼する。フィルタモジュールは、受信した情報を記録し、監査基準に照らして、正当なアクセスであるかを監査する。この際、フィルタモジュールがコマンド受信領域を読み込んで、ログ取得実施やアクセス制御の有無を制御することができる。監査基準に照らして、不正であると判断されたアクセスについては、本来の共有ライブラリへの呼出を行わずに、記録を残すのみとする、いわゆる共有ライブラリへのアクセスのフィルタリングを行うことができる。
【００２６】
以下、実施形態の詳細について説明する。
【００２７】
図１は、本実施形態における計算機のブロック例を示す図である。計算機１０１は、ＣＰＵ１０２、メモリ１０５、ストレージ１０３、通信装置１０６、キーボード、マウス又

はカード読取装置といった入力装置１０４、ディスプレイ又はプリンタといった出力装置１０７を有する。

【００２８】

図３に示すように、計算機１０１上で動作するソフトウェア、具体的にはＯＳ３０１、アプリケーション３０２、共有ライブラリ３０３等は、ストレージ１０３に格納されている。アプリケーション３０２や共有ライブラリ３０３は複数存在する場合が多い。ここで、ストレージ１０３は、ハードディスクドライブやＤＶＤドライブといった、不揮発性の記憶装置である。

【００２９】

これらのソフトウェアを実行する際、計算機１０１は、図２に示すように、ストレージ１０３よりＯＳ３０１、アプリケーション３０２及び共有ライブリ３０３を揮発性だが高速なメモリ１０５上に、ＯＳ２０１、アプリケーション２０２及び共有ライブリ２０３としてロードしてＣＰＵ１０２で実行する。以下、動作主体をソフトウェアで説明するが、実際は、そのソフトウェアを実行するＣＰＵ１０２によって動作は実行される。また、以下で説明されるソフトウェア間の通信は、指定された記憶領域にコマンドを記録することによるプログラム間通信によって行われる。

【００３０】

図６は、本実施形態における計算機１０１が有するソフトウェアのソフトウェア構成を示す図である。通常のアプリケーションプログラムは、アプリケーションが使用できる記憶領域（以下「ユーザモード領域」）６０１内に、アプリケーション１（６０３）やアプリケーション２（６０６）のように配置される。アプリケーションプログラム６０３、６０６は、コンパイル時または実行時に注入共有ライブラリ６０４、６０７と結合される。

【００３１】

注入共有ライブラリ６０４、６０７は、共有ライブラリ（６０５、６０８）とそれぞれ結合される。また、出力アプリケーション６０９も、ユーザモード領域６０１内に配置される。

ログ用共有領域６１０、コマンド受信領域６１１及び監査ポリシ６１２を有するフィルタモジュール６１３は、カーネル領域６０２に配置される。

【００３２】

アプリケーション１（６０３）、注入共有ライブラリ６０４、共有ライブラリ６０５及びフィルタモジュール６１３は同一の仮想空間に配置される。同様に、アプリケーション２（６０６）、注入ライブラリ６０７、共有ライブラリ６０８及びフィルタモジュール６１３は同一の仮想空間に、出力アプリケーション６０９とフィルタモジュール６１３は同一の仮想空間にそれぞれマップされる。アプリケーション１、アプリケーション２、出力アプリケーションは同時にマップされることはない。

【００３３】

アプリケーション１（６０３）およびアプリケーション２（６０６）はそれぞれ、注入対象となる共有ライブラリ６０５、６０８を呼び出す際には、図７に示す手順に従う。以下、図７に示された手順について説明する。尚、以下の説明では、アプリケーション１についてのみ説明するが、他のアプリケーションでも同様である。

【００３４】

まず、アプリケーション１は、共有ライブラリ６０５の外部公開関数を呼び出す（ステップ７０１）。

共有ライブラリ６０５への外部公開関数の呼び出しを受けたら、注入共有ライブラリ６０４は、その共有ライブラリ６０５への呼び出しをフックする。具体的なフック方法として、コンパイル時に注入共有ライブラリを静的にリンクする方法、動的リンクライブラリを注入ライブラリで置き換える方法、他アプリケーション空間で実行されているアプリケーションを用いて、目的とする外部公開関数へのアクセスをフックする方法などがある（ステップ７０２）。

【００３５】

外部公開関数の呼出をフックした注入共有ライブラリ６０４は、フィルタモジュール６１３をシステムコールで呼び出し、フックした呼出に含まれる共有ライブラリ呼出情報をフィルタモジュール６１３に送信する。共有ライブラリ呼出情報については後述する（ステップ７０３）。

【００３６】

共有ライブラリ呼出情報を受信したフィルタモジュール６１３は、コマンド受信領域６１１を参照し、出力アプリケーションからのコマンドを読み込む。ここで参照されたコマンドは、後述のステップで評価され、フィルタモジュール６１３の動作を決定することに使用される（ステップ７０４）。

【００３７】

フィルタモジュール６１３は、コマンド受信領域６１１から取得したコマンドに基づいて、ログ用共有領域６１０に共有ライブラリ呼出情報を記録するか否かを決定する。尚、ログ用共有領域６１０には、アプリケーション毎に共有ライブラリ呼出情報が格納される。したがって、アプリケーションごとの排他制御を行わずにすむ。さらに、ログ用共有領域６１０は、計算機が複数のプロセッサを有する場合に、そのプロセッサごとに区分けされていても良い。

【００３８】

取得されるコマンドには、呼び出しを行ったアプリケーションの種類、アプリケーションを使用しているユーザ、アプリケーションが呼び出した外部公開関数の種類など、共有ライブラリ呼出情報をログ用共有領域６１０に記録する際の条件が含まれていてもよい。この場合、フィルタモジュール６１３は、コマンドに含まれる条件に従って、共有ライブラリ呼出情報の記録の是非を決定する（ステップ７０５）。

【００３９】

ステップ７０５でログ取得が必要と判断された場合、フィルタモジュール６１３は、ログ用共有領域６１０のアプリケーション１の専用領域に対し、共有ライブラリ呼出情報を出力する（ステップ７０６）。

【００４０】

その後、フィルタモジュール６１３は、コマンド受信領域６１１から取得したコマンドに基づいて、共有ライブラリへの外部公開関数の呼出を監査するか否かを決定する。コマンド受信領域６１１から取得したコマンドには、呼び出しを行ったアプリケーションの種類、アプリケーションを使用しているユーザ、アプリケーションが呼び出した外部公開関数の種類など、呼出の監査を実行するための条件が含まれていてもよい。この場合、フィルタモジュール６１３は、コマンドに含まれえている条件に基づいて、呼出の監査の是非を決定する（ステップ７０７）。

【００４１】

ステップ７０７で共有ライブラリ呼出の監査が必要と判断された場合、フィルタモジュール６１３は、後述する監査ポリシ６１２に基づき、注入共有ライブラリ６０４からフィルタモジュール６１３に送信された共有ライブラリ呼出情報が、正当な呼出であるかを監査する（ステップ７０８）。

【００４２】

その後、フィルタモジュール６１３は、ステップ７０８で行われた共有ライブラリ呼出監査の結果を判定する（ステップ７０９）。

【００４３】

呼出が許可された場合又はステップ７０７で共有ライブラリ呼出監査が不要とされた場合、フィルタモジュール６１３からその通知を受けた注入共有ライブラリ６０４は、共有ライブラリ６０５への外部公開関数の呼出を行う。注入共有ライブラリ６０４は、共有ライブラリ６０５内の特定の外部公開関数に対し注入を行っており、注入共有ライブラリ６０４でフックした共有ライブラリ６０５の外部公開関数に送信すべき関数の引数も、アプリケーション１から共有ライブラリ呼出情報として受け取っている。したがって、共有注入ライブラリ６０４は、これら引数等の情報を利用して、対象となる共有ライブラリ６０

５の外部公開関数を、適切な引数を伴って呼出すことが可能である。

【００４４】

共有ライブラリ６０５から外部公開関数を呼び出した共有注入ライブラリ６０４は、その結果をアプリケーションに送信する。

【００４５】

共有ライブラリ呼出の監査の結果、呼出が許可されなかった場合、フィルタモジュール６１３は、ログ用共有領域６１０に不許可となった呼出の種類、不許可となった理由などを記録した上でアプリケーションにエラーを返す。エラーを返す他の方法として、共有ライブラリ６０５の呼出は行わず、アプリケーションに正常終了を返す方法、フィルタモジュール６１３が、許可されなかった呼出を共有ライブラリ６０５への呼出を許可される形に補正、すなわち呼出に使用される引数を許可される値に変更するといった補正を行ったうえで、注入共有ライブラリが共有ライブラリ呼出を行う方法などがある（ステップ７１１）。

【００４６】

図８は、ステップ７０３で、注入共有ライブラリがフィルタモジュールに対して送信する共有ライブラリ呼出情報の例を示す図である。共有ライブラリ呼出情報には、共有ライブラリ呼出元であるアプリケーション６０３の名前を示すアプリケーション名８０１、アプリケーション６０３のストレージ１０３に作成されたファイルシステム上の格納位置を示すアプリケーションパス８０２、アプリケーションを実行しているユーザを示すユーザ８０３、アプリケーションを実行しているユーザの所属するグループを示すグループ８０４、共有ライブラリ呼出時刻を示す呼出時刻８０５、アプリケーション内呼出位置を示す呼出元アドレス８０６、アプリケーションが共有ライブラリ呼出の際渡した引数の数を示す引数の個数８０７及びアプリケーションが共有ライブラリ呼出の際渡した引数の内容を示す引数のリスト８０８が含まれる。

【００４７】

図９は、監査ポリシ６１２に格納され、共有ライブラリ呼出の監査に用いられる監査ポリシの一例を示す図である。この監査ポリシは、少なくとも注入対象となる共有ライブラリの数だけ用意される。

【００４８】

列９０１は、監査対象となる共有ライブラリの外部公開関数名が登録される列である。以下の列に指定される項目は、列９０１で指定される関数への呼出を許可する条件を構成する。

【００４９】

列９０２は、列９０１で指定された関数への呼出を許可するアプリケーションの名前が登録される列である。登録される情報として、特定のアプリケーション名の他に、アプリケーション群を示す記号などを用いることで、アクセスを許可するアプリケーションの表記を簡略化する場合もある。

【００５０】

列９０３は、列９０１で指定された関数への呼出が許可されるアプリケーションが保存されるファイルシステム上の位置を表すアプリケーションパスが登録される列である。列９０２に登録された情報と組み合わせて使用することで、特定のパスに配置された任意のアプリケーションを指定し、アクセスを許可をすることができる。

【００５１】

列９０４は、列９０１で指定された関数への呼出が許可されるユーザを示す情報が登録される列である。登録される情報として、特定のユーザ名の他に、ユーザ郡を示す記号などを用いることで、アクセスを許可するユーザの表記を簡略化する場合もある。

【００５２】

列９０５は、列９０１で指定された関数への呼出を許可するユーザが所属するグループを示す情報が登録される列である。列９０４に登録された情報と組み合わせることによって、特定グループに所属する任意のユーザといったアクセス許可の仕方が可能となる。

【００５３】
列９０６は、列９０１で指定した関数への呼出が許可される時刻や時間帯を示す情報が登録される列である。
【００５４】
列９０７は、列９０１で指定した関数への呼出が許可されるアドレスを示す情報が登録される列である。特定のアドレス範囲内からの呼出を受け付けなくする指定や、コード領域からの呼出のみ許可する、スタック領域からの呼出を不許可とするといった指定が可能である。
【００５５】
列９０８は、列９０１で指定された関数への呼出を監査する際に、引数の内容まで監査する場合の監査方法が登録される列である。例えば、第１引数に８０という値が入っていなかった場合、アクセスを不許可とするといった監査内容が記載される。監査方法の他の実施形態として、アクセス監査モジュールを外付け、即ち、注入共有ライブラリ６０４にアクセス監査モジュールを機能的に追加できるようにしておき、監査ポリシの列９０８でアクセス監査モジュールの指定を行うといった方法もある。
尚、監査条件は、必要に応じて追加されても良い。
【００５６】
監査ポリシを用いた監査では、９０１で指定された関数に対し、９０２～９０８で示された監査条件をすべて満たした場合のみ、アプリケーションの共有ライブラリの呼出が許可される。
【００５７】
本実施形態の計算機では、監査ポリシ６１２等のフィルタモジュール６１３の設定の変更が出力アプリケーション６０９を用いて行われる。具体的には、フィルタモジュール６１３の挙動を変更する場合、管理者は出力アプリケーション６０９を用い、コマンド受信領域６１１へ変更すべき情報の書き込みを行う。たとえばシステム運用中に、監査ポリシ６１２を更新する場合の手順について、図１０を用いて説明する。
【００５８】
管理者は監査ポリシ変更の指示を出力アプリケーション６０９に入力する（ステップ１００１）。管理者から監査ポリシ変更指示を受けた出力アプリケーション６０９は、フィルタモジュール６１３のコマンド受信領域６１１に対し、監査ポリシ変更コマンドを出力する（ステップ１００２）。その後、フィルタモジュール６１３は出力アプリケーション６０９によって出力された管理ポリシ変更コマンドをコマンド受信領域６１１から読み取り、コマンドに従って監査ポリシ６１２を変更する（ステップ１００３）。
【００５９】
尚、管理者は、上記で示した管理ポリシ６１２の変更の指示の他にも、フィルタモジュール６１３の動作開始や停止などの指示を出力アプリケーション６０９を用いて行うことができる。
【００６０】
次に、本実施形態を計算機のセキュリティを向上するソフトウェアに適用した例について説明する。
図１１は、カーネル領域６０２にセキュリティモジュールを追加して、セキュリティを向上させるソフトウェアに対し、本実施形態を適用した場合のソフトウェア構成例を示すブロック図である。本来、セキュリティモジュール１１０７はセキュリティを保つためカーネル領域６０２に常駐している必要がある。しかし、セキュリティモジュール１１０７は、カーネル領域６０２内に存在するためには一般的なカーネルモジュールと同じ機能を備える必要があるため、悪意あるアプリケーションプログラム１１０３が適切な権限を取得することができれば、共有ライブラリ６０５を呼び出して、セキュリティモジュール１１０７をアンロードするまたは不活性化するといった無効化することができるという問題点がある。
【００６１】

しかし、図１１に示すように本実施形態を適用することで、矢印１１０８に示すように、悪意あるアプリケーション１１０３が共有ライブラリ６０４にアクセスする前に注入共有ライブラリ６０４によって、フィルタリングモジュール６０７にアクセスが通知される。更に、フィルタリングモジュール６０７が、自身に設定されている監視ポリシに基づいて、セキュリティモジュール１１０７の無効化を許可しないように動作する。これにより、意図しないセキュリティモジュールの無効化を防ぐことができる。

【００６２】

上述したように、フィルタモジュールをカーネル領域に設けることで、注入共有ライブラリからの共有ライブラリへのアクセス要求を監査する等の要求が、フィルタモジュールに対する一種のシステムコールになるので、計算機にロック機構を導入する必要が無い。又、カーネル領域においてアプリケーション毎にログを管理するので、処理情報の一貫性を保つことができる。又、ログ情報の取得の際にシステムコールを用いることで、処理速度が向上する。

【図面の簡単な説明】

【００６３】

【図１】計算機のハードウエア構成の例を示す図である。

【図２】メモリの内容例を示す図である。

【図３】ストレージの内容例を示す図である。

【図４】計算機上の仮想空間レイアウト例を示す図である。

【図５】計算機上の一仮想空間上のモジュール配置例を示す図である。

【図６】計算機上の仮想空間レイアウト例を示す図である。

【図７】共有ライブラリ呼出ログ取得および共有ライブラリ呼出監査手順をしめすフローチャートである。

【図８】共有ライブラリ呼出情報例を示す図である。

【図９】監査ポリシ例を示す図である。

【図１０】出力アプリケーションが、コマンド受信領域に監査ポリシ変更コマンドを出力する手順をしめすフローチャートである。

【図１１】計算機上の仮想空間レイアウト例を示す図である。

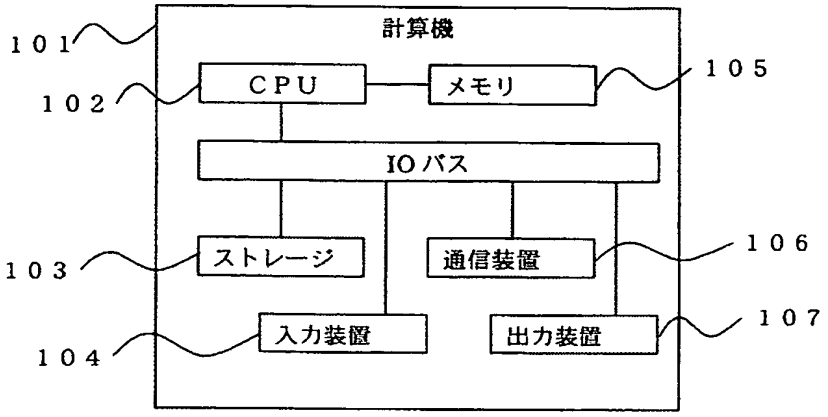
【符号の説明】

【００６４】

１０１…計算機、１０２…ＣＰＵ、１０３…ストレージ、１０４…入力装置、１０５…メモリ、１０６…通信装置、１０７…出力装置。

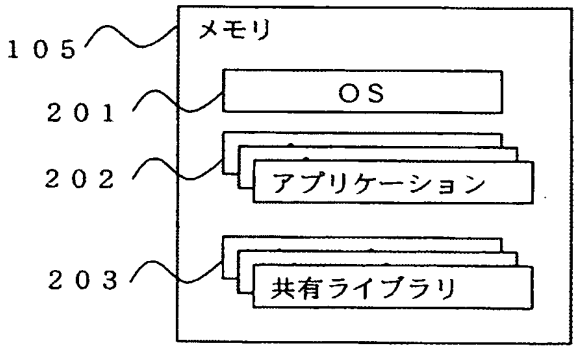
【書類名】図面
【図１】

図１



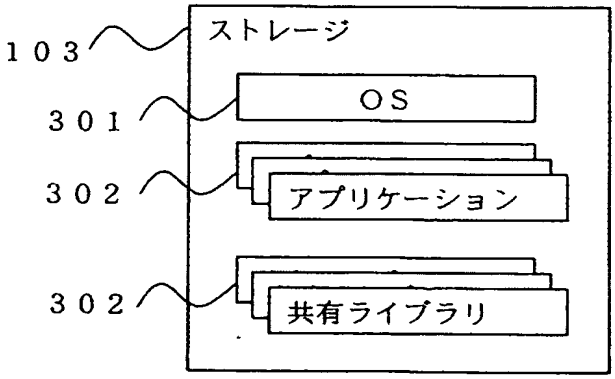
【図２】

図２



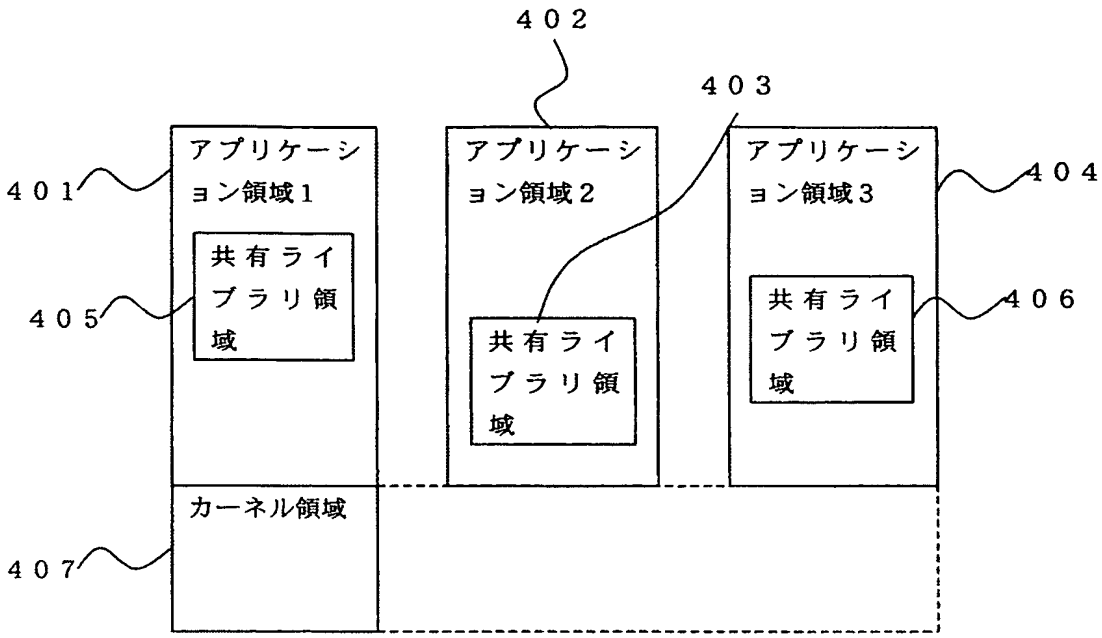
【図３】

図３



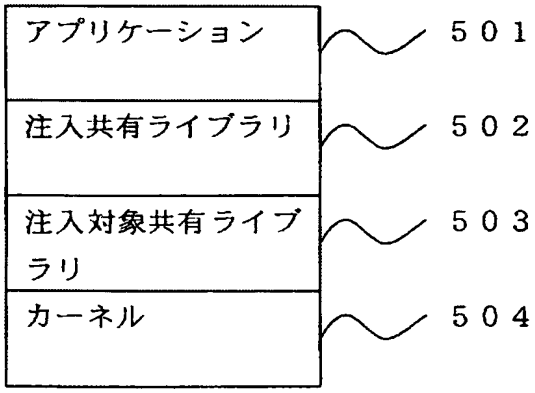
【図４】

図４


４０２
４０３
アプリケーション領域１
アプリケーション領域２
アプリケーション領域３
４０１
４０４
共有ライブラリ領域
４０５
共有ライブラリ領域
４０６
共有ライブラリ領域
カーネル領域
４０７

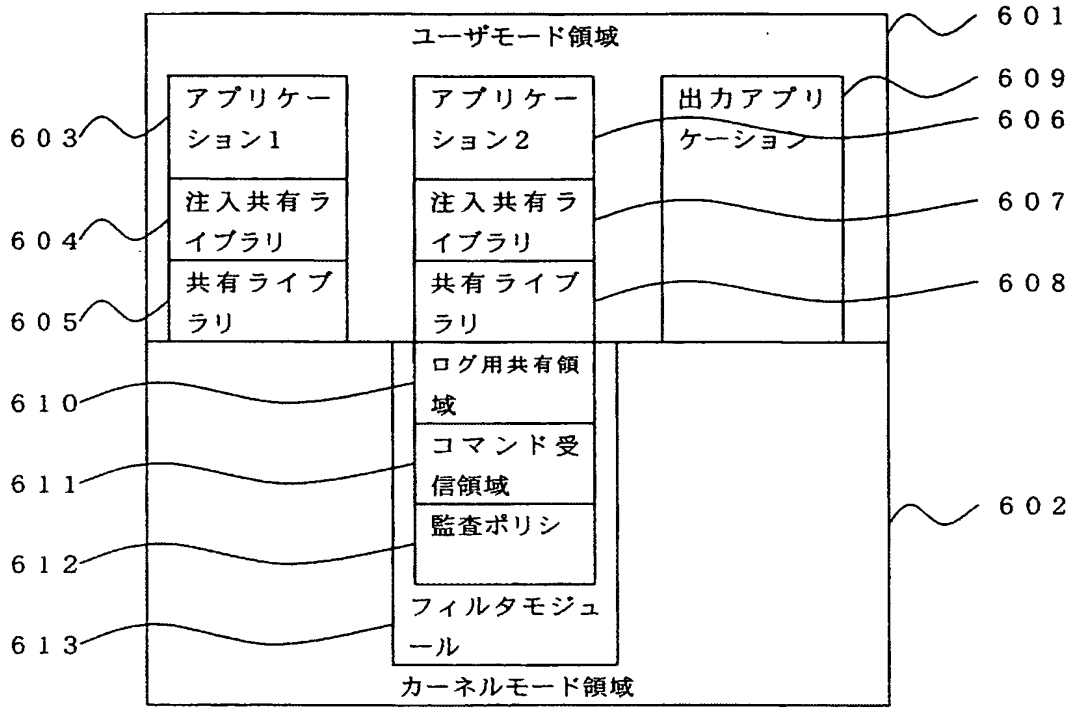
【図５】

図５


アプリケーション
５０１
注入共有ライブラリ
５０２
注入対象共有ライブラリ
５０３
カーネル
５０４

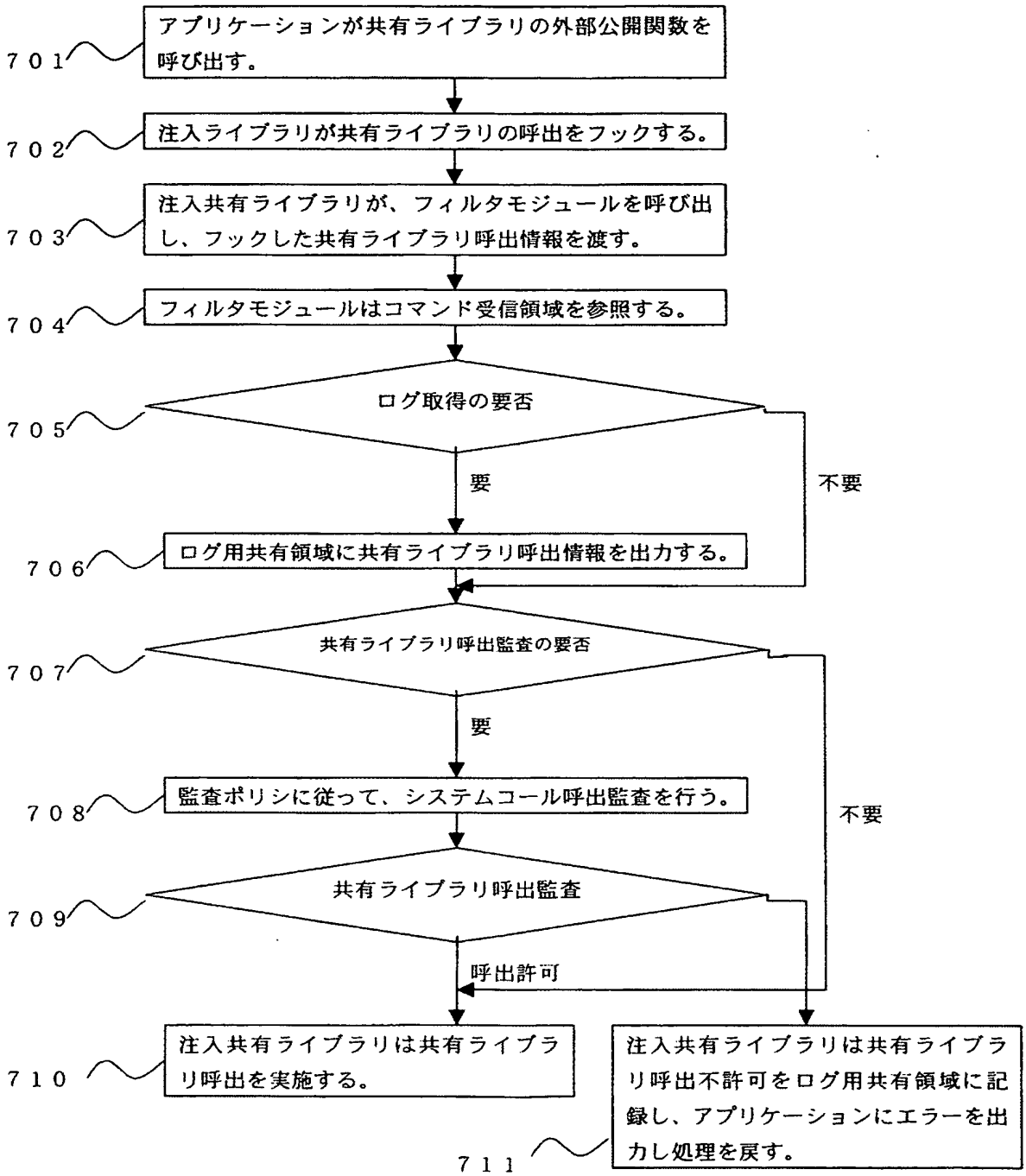
【図６】

図６


ユーザモード領域
アプリケーション1
注入共有ライブラリ
共有ライブラリ
アプリケーション2
注入共有ライブラリ
共有ライブラリ
出力アプリケーション
ログ用共有領域
コマンド受信領域
監査ポリシ
フィルタモジュール
カーネルモード領域
601
602
603
604
605
606
607
608
609
610
611
612
613

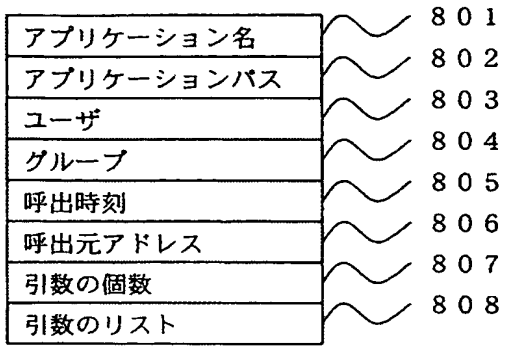
【図７】

図７


７０１
アプリケーションが共有ライブラリの外部公開関数を呼び出す。
７０２
注入ライブラリが共有ライブラリの呼出をフックする。
７０３
注入共有ライブラリが、フィルタモジュールを呼び出し、フックした共有ライブラリ呼出情報を渡す。
７０４
フィルタモジュールはコマンド受信領域を参照する。
７０５
ログ取得の要否
要
不要
７０６
ログ用共有領域に共有ライブラリ呼出情報を出力する。
７０７
共有ライブラリ呼出監査の要否
要
不要
７０８
監査ポリシに従って、システムコール呼出監査を行う。
７０９
共有ライブラリ呼出監査
呼出許可
７１０
注入共有ライブラリは共有ライブラリ呼出を実施する。
７１１
注入共有ライブラリは共有ライブラリ呼出不許可をログ用共有領域に記録し、アプリケーションにエラーを出力し処理を戻す。

【図８】

図８



【図９】

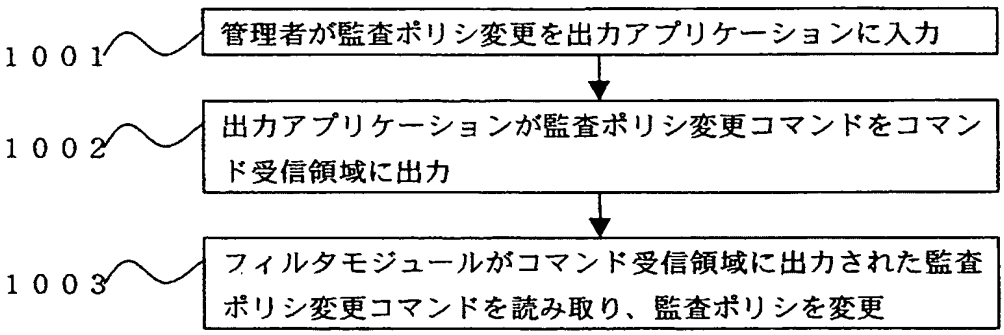
図９

９０１ 関数名	９０２ アプリケーション名	９０３ アプリケーションパス	９０４ ユーザ	９０５ グループ	９０６ 許可時刻	９０７ 許可アドレス	９０８ 引数監査内容
open	App1	/bin/	User1	Grp1	8:00-17:00	-7000000	
open	App2	/usr/bin/	User2	Grp2	24h	2000-4000	
close	App3	/var/bin/	User3	Grp3	17:00-0:00	80000000-	

９０９
９１０
９１１
９１２

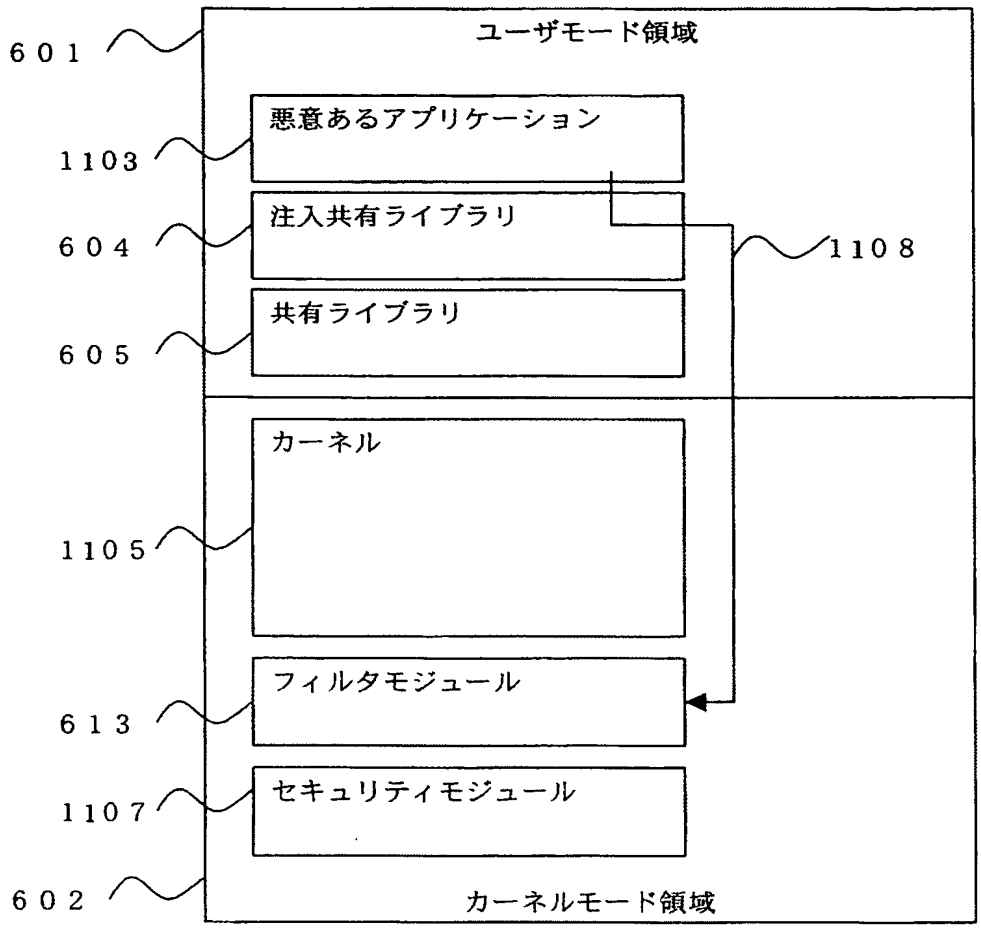
【図１０】

図１０



【図１１】

図１１


ユーザモード領域
６０１
悪意あるアプリケーション
１１０３
注入共有ライブラリ
６０４
１１０８
共有ライブラリ
６０５
カーネル
１１０５
フィルタモジュール
６１３
セキュリティモジュール
１１０７
６０２
カーネルモード領域

【書類名】要約書
【要約】
【課題】
共有ライブラリを介した外部公開関数への呼び出しを効率よく監査したい。
【解決手段】
共有ライブラリの外部公開関数をフックする注入共有ライブラリを用いてアプリケーションの共有ライブラリ呼出をフックし、カーネル領域に配置したフィルタモジュールで、共有ライブラリ呼出情報の記録、共有ライブラリ呼出の監査を行うことによって、共有ライブラリの変更無しに共有ライブラリ呼出をフィルタリングし、高速に一貫性のある記録を作成する。
【選択図】　図６

認定・付加情報

特許出願の番号	特願２００３－３８６０９１
受付番号	５０３０１８９１７２６
書類名	特許願
担当官	第七担当上席　　００９６
作成日	平成１５年１１月１８日

＜認定情報・付加情報＞

【提出日】　平成15年11月17日

特願２００３－３８６０９１

出　願　人　履　歴　情　報

識別番号　　［０００００５１０８］

１．変更年月日　　１９９０年　８月３１日
［変更理由］　　新規登録
住　所　　東京都千代田区神田駿河台４丁目６番地
氏　名　　株式会社日立製作所